CiteSeer Find: multiple loop and instruction Documents Citations

Searching for **multiple loop and instruction**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Amazon B&N Google (RI) Google (Web) CSB DBLP
20 documents found. **Order: number of citations.**

Effective Compiler Support for Predicated Execution .. - Mahlke, Lin, Chen, .. (1992) (Correct) (109 citations)
For numeric code, overlapping the execution of **multiple loop** it erations using software pipeline refers to the conditional execu tion of **instructions** based on the value of a boolean source predicate. When the predicate has value T, the **instruction** is executed normally and when the predicate
cardit.et.tudelft.nl/~steven/ilp/mahlke92.ps.gz

Automatic Program Parallelization - Banerjee, Eigenmann, Nicolau (1993) (Correct) (70 citations)
concepts are meaningful only in the case of **multiple loops** and they are defined for dou ble loops. to the most recent literature and discusses both **instruction**level and coarsegrain paral 1 lelization
www.csrd.uiuc.edu/reports/1250.ps.gz

An Affine Partitioning Algorithm to Maximize Parallelism and.. - Amy Lim (1999) (Correct) (4 citations)
op timize parallelism and communication across **multiple loop** nests. For example, Anderson and Lam's of an affine partition, which maps instances of **instructions** into the time or processor space via an affine S ffi DF j ?where ffi S is the set of **instructions**. An **instruction** is an indi visible unit such
www-suif.stanford.edu/papers/lim99.ps

Compiler Analysis to Implement Point-to-Point Synchronization in.. - Nguyen (1993) (Correct) (3 citations)
www.lcs.mit.edu/publications/pubs/pdf/MIT-LCS-TR-595.pdf

Determining Asynchronous Pipeline Execution Times - Donaldson, Ferrante (1996) (Correct) (2 citations)
16]where assembly language **instructions** from **multiple loop** iterations are scheduled to reduce the number which is being pipelined. Examples include **instruction** and vector pipelining in hardware [14]and pipelining [1, 16]where assembly language **instructions** from **multiple loop** iterations are scheduled to
www-cse.ucsd.edu/users/vdonalds/tr481.ps

Handling Irreducible Loops: Optimized Node Splitting vs.. - Unger, Mueller (2001) (Correct) (1 citation)
the sink of the backedge(s) of such a **loop. Multiple** entry loops cause irreducible regions of recent VLIW-like architectures highly rely on **instruction** scheduling. The contributions of this paper transformations and optimizations to exploit **instruction**-level parallelism cannot be applied to such
www.informatik.hu-berlin.de/~mueller/ftp/pub/mueller/papers/europar01.ps.gz

Flattening VLIW code generation for imperfectly nested loops - Peter Knijnenburg.. (1998) (Correct) (1 citation)
consists of converting an imper fectly nested **multiple loop** nest into a single loop. For a target VLIW for (im)perfectly nested loops for Very Long **Instruction** Word (VLIW) architectures. The transformation for (im)perfectly nested loops for Very Long **Instruction** Word (VLIW) architec tures. In the last year
www.wi.leidenuniv.nl/~peterk/flatten.ps.gz

Dynamic Vectorization: A Mechanism for Exploiting.. - Vajapeyam, Joseph, Mitra (1999) (Correct) (1 citation)
body in vector form in the **instruction** window. **Multiple loop** iterations are issued from the single copy of a tool for quickly building up a large logical **instruction** window. Dynamic vectorization converts vectorization converts repetitive dynamic **instruction** sequences into vector form, enabling the
www.ecsl.cs.sunysb.edu/~tulika/dv.ps.Z

Acceleration of First and Higher Order Recurrences on - Processors With Instruction (Correct)
of a single innermost loop to a complex **multiple loop** configuration may be very difficult or and Higher Order Recurrences on Processors with **Instruction** Level Parallelism Michael Schlansker and a broad class of recurrences on processors with **instruction** level parallelism. We introduce a new
www.hpl.hp.com/research/itc/car/papers/../papers/Acceleration.pdf

Runtime Code Parallelization for On-Chip Multiprocessors - Kandemir Zhang Cse (2003) (Correct)
if the application being optimized consists of **multiple loop** nests, each loop nest can demand a different

Each processor is equipped with data and **instruction** caches and can operate independently i.e.it
In this work, we focus on processors, data and **instruction** caches, and the shared memory. The scope of
www.cse.psu.edu/~wzhang/paper/DATE2003_1.pdf

Characterizing Coarse-Grained Reuse of Computation - Subramanya Sastry Rastislav (Correct)
www.ece.wisc.edu/~jes/papers/fddo00.sastry.ps

Performance of a Micro-threaded Pipeline - Luo, Jesshope (Correct)
www.jrpit.flinders.edu.au/confpapers/CRPITV6Luo.pdf

Runtime Predictability of Loops - de Alba, Kaeli (2001) (Correct)
loop execution history and allows us to predict **multiple loop** iterations dynamically. 1 Introduction
architectures, a large window of valid **instructions** must be available. While researchers have been
there still remains the need for more aggressive **instruction** delivery. Loop bodies possess a large amount
www.ece.neu.edu/students/mdealba/wwc4_loops.ps

Explicit Dynamic Scheduling: A Practical Micro-Dataflow.. - Beckmann.. (1993) (Correct)
is especially effective in inner loops, where **multiple loop** iterations can be overlapped, if necessary,
the performance advantages of dataflow at the **instruction** level. EDS unifies pipeline and memory latency
superpipelined RISC processors relying on static **instruction** scheduling. Research supported primarily by
ftp.csrd.uiuc.edu/pub/misc/beckmann/csrd.1316.ps.gz

Analyzing Asynchronous Pipeline Schedules - Donaldson, Ferrante (1997) (Correct)
(6,7) where assembly language **instructions** from **multiple loop** iterations are scheduled to reduce the number
which is being pipelined. Examples include **instruction** and vector pipelining in hardware, 5) and
pipelining ,6,7) where assembly language **instructions** from **multiple loop** iterations are scheduled to
www.cs.ucsd.edu/users/ferrante/valijpp.ps

Experiments with Data Layouts - Kandemir, Choudhary, Shenoy.. (1997) (Correct)
a single loop nest whereas the others contain **multiple loop** nests. For each program in the suite, we
Hp Parisc 8000 Processors With 1mb Data And 1mb **Instruction** Cache. The Exemplar Is Built Around The
gate.ee.lsu.edu/pub/jxr/papers/tr/cpdc976.ps

Final Report on Research in Parallel Computing.. - December Carnegie (1996) (Correct)
produces a schedule so that the executions of **multiple loop** iterations are overlapped. Operations from
loop iterations are either placed into the same **instruction** word or are interleaved, depending on the type
or are interleaved, depending on the type of **instruction**level parallelism provided in the target
reports-archive.adm.cs.cmu.edu/anon/1996/CMU-CS-96-100E.ps

Modelling Instruction-Level Parallelism for Software.. - Ali-Reza Adl-Tabatabai (Correct)
Modelling **Instruction**-Level Parallelism for Software Pipelining
www.cs.cmu.edu/afs/cs/user/gyl/www/ifip93.ps

Search Results

Search Results for: **[single instruction<AND>((multiple<AND>(((loop instruction)))))]**
Found **17** of **129,310 searched.**

Search within Results

1 OHMEGA: a VLSI superscalar processor architecture for numerical applications 84%

Masaitsu Nakajima , Hiraku Nakano , Yasuhiro Nakakura , Tadahiro Yoshida , Yoshiyuki Goi , Yuji Nakai , Reiji Segawa , Takeshi Kishida , Hiroshi Kadota

ACM SIGARCH Computer Architecture News , Proceedings of the 18th annual international symposium on Computer architecture April 1991
Volume 19 Issue 3

2 An evaluation of branch architectures 80%

J. A. DeRosa , H. M. Levy

Proceedings of the 14th annual international symposium on Computer architecture June 1987

Branch instructions form a significant fraction of executed instructions, and their design is thus a crucial component of any architecture. This paper examines three alternatives in the design of branch instructions: delayed vs. non-delayed branches, one- vs. two-instruction branches, and the use or non-use of condition codes. Simulation and analytical techniques are used to provide quantitative comparisons between these choices.

3 Software and hardware parallelism on the iWarp multi-computer 80%

Herbert G. Mayer , Brent Baxter

Proceedings of the 5th international conference on Supercomputing June 1991

4 Superscalar architectures: Select-free instruction scheduling logic 77%

Mary D. Brown , Jared Stark , Yale N. Patt

Proceedings of the 34th annual ACM/IEEE international symposium on Microarchitecture December 2001

Pipelining allows processors to exploit parallelism. Unfortunately, critical loops---pieces of logic that must evaluate in a single cycle to meet IPC (Instructions Per Cycle) goals---prevent deeper pipelining. In today's processors, one of these loops is the instruction scheduling (wakeup and select) logic [10]. This paper describes a technique that pipelines

this loop by breaking it into two smaller loops: a critical, single-cycle loop for wakeup; and a non-critical, potentially multi-cycle, lo ...

5 Discovering machine-specific code improvements 77%

Peter B. Kessler

ACM SIGPLAN Notices , Proceedings of the 1986 SIGPLAN symposium on Compiler contruction July 1986

Volume 21 Issue 7

I have designed and built a compiler construction tool that automates much of the case analysis necessary to exploit special purpose instructions on a target machine. Given a suitable description of the target machine, my analysis identifies instruction sequences that are equivalent to single instructions. During code generation, these equivalences can be used to avoid inefficient instruction sequences in favor of more efficient instructions.I present a working prototype of the i ...

6 Compiler scheduling: Reduced code size modulo scheduling in the absence of hardware support 77%

Josep Llosa , Stefan M. Freudenberger

Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture November 2002

Modulo scheduling is a very effective instruction scheduling technique that exploits Instruction Level Parallelism (ILP) in loop bodies by overlapping the execution of successive iterations. Unfortunately, modulo scheduling has been shown to cause heavy code expansion. To avoid the penalties of code expansion, some processors have dedicated hardware support for modulo scheduled loops. However, this dedicated hardware support has a cost in chip area, cycle time, processor complexity, and compiler ...

7 A cellular general purpose computer 77%

R. G. Cornell , H. C. Torng

ACM SIGARCH Computer Architecture News , Proceedings of the 2nd annual symposium on Computer architecture December 1974

Volume 3 Issue 4

A 2-dimensional cellular general-purpose computer is specified. This particular cellular computer is distinguished from previously proposed, locally-controlled cellular computers in that the cellular structure is "hidden" from the user. At the ISP level, the machine is similar to a small-scale computer of the von Neumann type. However, the architecture of the computer does not feature physically isolated functional units to implement memory, processor, or control. As a result, we present a machi ...

8 Application specific compiler/architecture codesign: a case study 77%

Oliver Wahlen , Tilman Glökler , Achim Nohl , Andreas Hoffmann , Rainer Leupers , Heinrich Meyr

ACM SIGPLAN Notices , Proceedings of the joint conference on Languages, compilers and tools for embedded systems: software and compilers for embedded systems June 2002

Volume 37 Issue 7

This paper proposes an architecture exploration methodology for application specific instruction set processors (ASIPs) including a C compiler and a VHDL model in the exploration loop. For a given application the target architecture is an instance of the scalable ALICE VLIW architecture which will be presented in this paper. In a case study it will be explained how the LISA processor design platform in conjunction with the CoSy compiler environment significantly reduces the time for exploration ...

9 Energy aware compilation for DSPs with SIMD instructions 77%

Markus Lorenz , Lars Wehmeyer , Thorsten Dräger

ACM SIGPLAN Notices , Proceedings of the joint conference on Languages, compilers and tools for embedded systems: software and compilers for embedded systems June 2002

Volume 37 Issue 7

The growing use of digital signal processors (DSPs) in embedded systems necessitates the use of optimizing compilers supporting special hardware features. In this paper we present compiler optimizations with the aim of minimizing energy consumption of embedded applications: This comprises loop optimizations for exploitation of SIMD instructions and zero overhead hardware loops in order to increase performance and decrease the energy consumption. In addition, we use a phase coupled code generator ...

10 Flow-control machines: the structured execution architecture (SXA) 77%

J. M. Terry

ACM SIGARCH Computer Architecture News September 1987

Volume 15 Issue 4

Can the looping, branching, sequencing, and modularity of structured programming be implemented effectively by single instructions at the machine-instruction level? A proposal for a class of machines to do so is presented. Flow control verbs such as REPEAT and IF/THEN/ELSE are represented in large-format instructions containing pointers to conditional controls and other instructions. An active-instruction stack is used to nest flow structures. Independent buses for logically distinct memory spac ...

11 Algorithm and architecture of a 1V low power hearing instrument DSP 77%

Finn Møller , Nikolai Bisgaard , John Melanson

Proceedings of the 1999 international symposium on Low power electronics and design August 1999

12 Vector architectures: past, present and future 77%

Roger Espasa , Mateo Valero , James E. Smith

Proceedings of the 12th international conference on Supercomputing July 1998

13 Speculative multithreaded processors 77%

Pedro Marcuello , Antonio González , Jordi Tubella

Proceedings of the 12th international conference on Supercomputing July 1998

14 A compilation technique for software pipelining of loops with conditional jumps 77%

Kemal Ebcioğlu

Proceedings of the 20th annual workshop on Microprogramming December 1987

We describe a compilation algorithm for efficient software pipelining of general inner loops, where the number of iterations and the time taken by each iteration may be unpredictable, due to arbitrary if-then- else statements and conditional exit statements within the loop. As our target machine, we assume a wide instruction word architecture that allows multi-way branching in the form of if-then-else trees, and that allows conditional register transfers depending on where the microinstruct ...

15 Techniques for extracting instruction level parallelism on MIMD architectures 77%

Gary Tyson , Matthew Farrens

Proceedings of the 26th annual international symposium on Microarchitecture

December 1993

16 Implementing signatures for C++ 77%

Gerald Baumgartner , Vincent F. Russo
ACM Transactions on Programming Languages and Systems (TOPLAS) January 1997
Volume 19 Issue 1

We outline the design and detail the implementation of a language extension for abstracting types and for decoupling subtyping and inheritance in C++. This extension gives the user more of the flexibility of dynamic typing while retaining the efficiency and security of static typing. After a brief discussion of syntax and semantics of this language extension and examples of its use, we present and analyze three different implementation techniques: a preprocessor to a C++ compiler, an implem ...

17 Compiler transformations for high-performance computing 77%

David F. Bacon , Susan L. Graham , Oliver J. Sharp
ACM Computing Surveys (CSUR) December 1994
Volume 26 Issue 4

In the last three decades a large number of compiler transformations for optimizing programs have been implemented. Most optimizations for uniprocessors reduce the number of instructions executed by the program using transformations based on the analysis of scalar quantities and data-flow techniques. In contrast, optimizations for high-performance superscalar, vector, and parallel processors maximize parallelism and memory locality with transformations that rely on tracking the properties o ...

Results 1 - 17 of 17 **short listing**

Your search matched **19** of **1015452** documents.
A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance** in **Descending** order.

Refine This Search:
You may refine your search by editing the current search expression or entering a new one in the text box.

multiple<and>loop<and>instruction<and>single Search

☐ Check to search within this result set

Results Key:
JNL = Journal or Magazine **CNF** = Conference **STD** = Standard

---

1 **A topological sorting and loop cleansing algorithm for a constrained MIMD compiler of shift-invariant flow graphs**
Lee, S.; Barnwell, T., III;
Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '86. , Volume: 11 , Apr 1986
Pages:2927 - 2930

[Abstract] [PDF Full-Text (176 KB)] **IEEE CNF**

---

2 **A 16-bit multiflow concurrent processor for VTR control**
Minakuchi, H.; Kunihira, T.; Ohta, Y.; Suehiro, K.; Soga, J.; Sakai, T.; Ochi, T.; Urade, M.; Okamoto, T.;
Consumer Electronics, IEEE Transactions on , Volume: 34 , Issue: 3 , Aug 1988
Pages:580 - 587

[Abstract] [PDF Full-Text (789 KB)] **IEEE JNL**

---

3 **Fast software implementation of MPEG advanced audio encoder**
Dimkoviae, I.; Milovanoviae, D.; Bojkoviae, Z.;
Digital Signal Processing, 2002. DSP 2002. 2002 14th International Conference on , Volume: 2 , 1-3 July 2002
Pages:839 - 843 vol.2

[Abstract] [PDF Full-Text (421 KB)] **IEEE CNF**

---

4 **Structure-based automatic extraction of the program heterogeneity**
Guosun Zeng; Xinda Lu; Jingcun Wang; Dingkang Zhou;
High Performance Computing in the Asia-Pacific Region, 2000. Proceedings. The Fourth International Conference/Exhibition on , Volume: 1 , 14-17 May 2000
Pages:261 - 262 vol.1

[Abstract] [PDF Full-Text (140 KB)] IEEE CNF

---

5 **Design of a low cost trainer for flow control**
Rehg, J.A.;
Frontiers in Education Conference, 1998. FIE '98. 28th Annual , Volume: 3 , 4-7 Nov. 1998
Pages:1063 - 1067 vol.3

[Abstract] [PDF Full-Text (420 KB)] IEEE CNF

---

6 **Improved instructional techniques in HVAC control systems**
Watton, A.; Marcks, R.K.; Solem, G.;
Frontiers in Education Conference, 1996. FIE '96. 26th Annual Conference., Proceedings of , Volume: 1 , 6-9 Nov. 1996
Pages:35 - 38 vol.1

[Abstract] [PDF Full-Text (584 KB)] IEEE CNF

---

7 **Cost-conscious strategies to increase performance of numerical programs on aggressive VLIW architectures**
Lopez, D.; Llosa, J.; Valero, M.; Ayguade, E.;
Computers, IEEE Transactions on , Volume: 50 , Issue: 10 , Oct. 2001
Pages:1033 - 1051

[Abstract] [PDF Full-Text (2488 KB)] IEEE JNL

---

8 **Loop coalescing and scheduling for barrier MIMD architectures**
O'Keefe, M.T.; Dietz, H.G.;
Parallel and Distributed Systems, IEEE Transactions on , Volume: 4 , Issue: 9 , Sept. 1993
Pages:1060 - 1064

[Abstract] [PDF Full-Text (396 KB)] IEEE JNL

---

9 **Optimizing loop performance for clustered VLIW architectures**
Yi Qian; Carr, S.; Sweany, P.;
Parallel Architectures and Compilation Techniques, 2002. Proceedings. 2002 International Conference on , 22-25 Sept. 2002
Pages:271 - 280

[Abstract] [PDF Full-Text (282 KB)] IEEE CNF

---

10 **Extracting SIMD parallelism from `for' loops**
Gustin, V.; Bulic, P.;
Parallel Processing Workshops, 2001. International Conference on , 3-7 Sept. 2001

Pages:23 - 28

[Abstract] [PDF Full-Text (360 KB)] IEEE CNF

---

11 **Special session on low-power systems on chips (SOCs)**
Piguet, C.; Renaudin, M.; Omnes, T.J.-F.;

Design, Automation and Test in Europe, 2001. Conference and Exhibition 2001. Proceedings , 13-16 March 2001
Pages:488 - 494

[Abstract] [PDF Full-Text (576 KB)] **IEEE CNF**

---

12 **Data speculative multithreaded architecture**
Marcuello, P.; Gonzalez, A.;
Euromicro Conference, 1998. Proceedings. 24th , Volume: 1 , 25-27 Aug. 1998
Pages:321 - 324 vol.1

[Abstract] [PDF Full-Text (376 KB)] **IEEE CNF**

---

13 **Conflict-free access to multiple single-ported register files**
Mueller, S.M.; Vishkin, U.;
Parallel Processing Symposium, 1997. Proceedings., 11th International , 1-5 April 1997
Pages:672 - 678

[Abstract] [PDF Full-Text (560 KB)] **IEEE CNF**

---

14 **GPMB-software pipelining branch-intensive loops**
Zhizhong Tang; Gang Chen; Chihong Zhang; Yingwei Zhang; Bogong Su; Habib, S.;
Microarchitecture, 1993. Proceedings of the 26th Annual International Symposium on , 1-3 Dec. 1993
Pages:21 - 29

[Abstract] [PDF Full-Text (592 KB)] **IEEE CNF**

---

15 **FORGE 90: a parallel programming environment**
Levesque, J.M.;
Compcon Spring '92. Thirty-Seventh IEEE Computer Society International Conference, Digest of Papers. , 24-28 Feb. 1992
Pages:291 - 294

[Abstract] [PDF Full-Text (288 KB)] **IEEE CNF**

---

1 2 Next

Home | Log-out | Journals | Conference Proceedings | Standards | Search by Author | Basic Search | Advanced Search | Join IEEE | Web Account | New this week | OPAC Linking Information | Your Feedback | Technical Support | Email Alerting | No Robots Please | Release Notes | IEEE Online Publications | Help | FAQ| Terms | Back to Top

Your search matched **19** of **1015452** documents.
A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance** in **Descending** order.

Refine This Search:
You may refine your search by editing the current search expression or entering a new one in the text box.

multiple<and>loop<and>instruction<and>single Search

☐ Check to search within this result set

Results Key:
JNL = Journal or Magazine **CNF** = Conference **STD** = Standard

---

16 **A neural network processor incorporating multiple on-chip cache memories**
Tay, O.N.; Noakes, P.D.;
Neural Networks, 1991. 1991 IEEE International Joint Conference on , 18-21 Nov. 1991
Pages:2222 - 2227 vol.3

[Abstract] [PDF Full-Text (356 KB)] **IEEE CNF**

---

17 **Data parallel computers and the FORALL statement**
Albert, E.; Lukas, J.D.; Steele, G.L., Jr.;
Frontiers of Massively Parallel Computation, 1990. Proceedings., 3rd Symposium on the , 8-10 Oct. 1990
Pages:390 - 396

[Abstract] [PDF Full-Text (480 KB)] **IEEE CNF**

---

18 **Compiling SIMD programs for MIMD architectures**
Quinn, M.J.; Hatcher, P.J.;
Computer Languages, 1990., International Conference on , 12-15 March 1990
Pages:291 - 296

[Abstract] [PDF Full-Text (440 KB)] **IEEE CNF**

---

19 **Experimental performance evaluation of the clustered multiprocessor system MUGEN**
Horiguchi, S.; Kawazoe, Y.;
TENCON '89. Fourth IEEE Region 10 International Conference , 22-24 Nov. 1989
Pages:205 - 208